WHAT IS CLAIMED IS:

1. A method of selecting certain portions of a computer program for compilation, the method comprising:

computing a compilation threshold corresponding to an execution frequency at which a decreasing hazard rate corresponds to a reciprocal of a break-even number of executions that recoup computational costs of compilation; and

during execution of the computer program, dynamically compiling individual ones of the portions based on correspondence between observed execution for the individual portions and the compilation threshold.

2. A method as recited in claim 1,

performing the computation of the compilation threshold coincident with execution of the computer program and using runtime information from the execution.

3. A method as recited in claim 1,

wherein the hazard rate, $hr(x)$, for a particular one of the computer program portions at least approximates a probability that the particular portion will stop being executed in the computer program given that the particular portion has executed x times.

4. A method as recited in claim 1, wherein the break-even number of executions is a function of:

time or other execution resource to execute an uncompiled version of a particular one of the computer program portions;

time or other execution resource to compile the particular portion; and

time or other execution resource to execute a compiled version of the particular portion.

5. A method as recited in claim 1,

wherein the portions correspond to functions, procedures, methods or routines of the computer program.

6. A method as recited in claim 1,

wherein the portions correspond to bytecodes executable in an execution environment.

7. A method as recited in claim 1,

wherein those computer program portions not selected for compilation are interpreted.

8. A method as recited in claim 1,

wherein a particular one of the computer program portions is interpreted for a first subset of executions thereof; and

wherein, after the dynamic compilation, subsequent executions are of a compiled version of the particular portion.

9. A method as recited in claim 1,

wherein a first subset of executions of a particular one of the computer program portions is of a first compiled version thereof;

wherein, after the dynamic compilation, subsequent executions are of a second compiled version of the particular portion.

10. A method as recited in claim 9,

wherein the second compiled version is substantially optimized as compared with the first compiled version.

11. An execution environment for a computer program encoded using execution codes that may optionally be executed in either first or second form, the execution environment comprising:

a dynamic compilation mechanism that transforms an implementation of a particular execution code to the second form thereof, wherein the

second form is substantially optimized as compared to the first form; and

for at least the particular execution code, an execution-time measurement of execution frequency at which a decreasing hazard rate corresponds to a reciprocal of a break-even number of executions that recoup computational costs of transformation to the second form,

wherein the dynamic compilation mechanism is responsive to the execution-time measurement.

12. An execution environment as recited in claim 11,

wherein the first and second forms respectively include uncompiled and compiled version of the execution code.

13. An execution environment as recited in claim 11,

wherein the first and second forms both include compiled versions of the execution code, but the second form is substantially optimized as compared to the first form.

14. An execution environment as recited in claim 11,

wherein the execution code is a bytecode.

15. A computer program product encoded in at least one computer readable medium, the computer program product comprising:

first instructions executable on a processor to instrument execution of a computer program executing thereon, the first instructions providing data indicative of execution frequency for at least a particular portion of the computer program; and

second instructions executable to identify a particular point in the execution of the computer program at which a decreasing hazard rate calculated from the execution frequency data for the particular portion of the computer program corresponds to a reciprocal of a break-even number of executions thereof that recoup computational costs of transformation to an optimized form.

16. The computer program product of claim 15,
wherein the particular portion is dynamically compiled to the optimized form coincident with identification of the particular point.

17. The computer program product of claim 15, further comprising:
a dynamic compiler.

18. The computer program product of claim 15, embodied as part of an execution environment for the computer program.

19. The computer program product of claim 15, embodied as part of the computer program.

20. The computer program product of claim 15,
wherein the at least one computer readable medium is selected from the set of a disk, tape or other magnetic, optical, or electronic storage medium and a network, wireline, wireless or other communications medium.

21. An apparatus comprising:
means for dynamically transforming an implementation of a particular execution code to an optimized form thereof; and
means for measuring execution frequency for at least the particular execution code and, based thereon, determining a point in an execution of computer code that includes the particular execution code at which a decreasing hazard rate corresponds to a reciprocal of a break-even number of executions that recoup computational costs of transformation to the optimized form.